

# Package ‘SVN’

April 10, 2019

**Type** Package

**Title** Statistically Validated Networks

**Version** 1.0

**Date** 2019-04-10

**Author** Damien Challet

**Maintainer** Damien Challet <damien.challet@gmail.com>

**Description** Determines networks of significant synchronization between the discrete states of nodes; see Tumminello et al <doi:10.1371/journal.pone.0017994>.

**License** GPL (>= 2.0)

**Depends** data.table, igraph, memoise

## Imports

**RoxygenNote** 6.1.0

**Encoding** UTF-8

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2019-04-10 18:25:30 UTC

## R topics documented:

SVN-package . . . . .	2
SVN_clusters . . . . .	3
SVN_links . . . . .	3

<b>Index</b>	<b>5</b>
--------------	----------

**Description**

Statistically validated networks are built from the states of nodes and from their curious (statistically speaking) synchronization: two nodes are linked if their states are anomalously synchronized. The associated null hypothesis is that the states of nodes are Poissonian processes, i.e., independent from each other.

**Details**

The main function `SVN_links` expects a matrix/data.frame/data.table of states, the line number being the index and while a given column corresponds to a given node, and outputs the results network in an igraph format. A Multiple Hypothesis Testing correction is applied since one performs several (rather, many) tests. It is then up to the user to use some network clustering technique if needed. For the convenience of the user, the function `SVN_clusters` applies the infoMap or Louvain method to the output of `SVN_links` and returns the resulting clustering. For small enough timeseries, it may be useful to use the options(`svn.memoise=TRUE`) so as to avoid expensive calls to the phyper function.

**Author(s)**

Maintainer: Damien Challet <damien.challet@gmail.com>

**References**

Tumminello, M., Micciche, S., Lillo, F., Piilo, J., & Mantegna, R. N. (2011). Statistically validated networks in bipartite complex systems. *PloS one*, 6(3), e17994.

**See Also**

**igraph**

**Examples**

```
x=sample(c(1,0),1000,replace = TRUE) # random vector of 0s and 1s
xx=x; x[1]=1-x[1] # one modifies just one element. x and xx are very similar
x_rev=rev(x) # x and x_rev are not
y=sample(c(1,0),1000,replace = TRUE) # y is another random vector of 0s and 1s
M=cbind(x,xx,x_rev,y) # builds the matrix of states
mylinks=SVN_links(M)
print(mylinks)

# one can compute clusters as well
myclusters=SVN_clusters(mylinks)
```

---

SVN_clusters	<i>Find clusters in Statistically Validated Networks</i>
--------------	--

---

**Description**

Find clusters in Statistically Validated Networks

**Usage**

```
SVN_clusters(links, cluster.method = "infomap")
```

**Arguments**

`links` an object obtained from the `SVN_links` function  
`cluster.method` a string, either "infomap" or "louvain", that selects the clustering method

**Value**

an igraph communities object

**Examples**

```
M=matrix(rbinom(200,size = 1,0.5),nrow=50)
mylinks=SVN_links(M)
# no links, then
print(mylinks)

# another example
x=c(1,0,0,0,0,0,0,1,1,1,1,0,0)
xx=c(1,1,0,0,0,0,0,1,1,1,1,0,0)
xrev=rev(x)
w=sample(x,length(x))
M=rbind(x,xx,xrev,w)
mylinks=SVN_links(M)

## if one wants clusters as well
myclusters=SVN_clusters(mylinks)
```

---

SVN_links	<i>Statistical validated networks</i>
-----------	---------------------------------------

---

**Description**

Statistical validated networks

**Usage**

```
SVN_links(states_vs_t, alpha = 0.01, MHT.correction = "bonferroni",
  exclude.states = NULL, states.pair.types = "all",
  alternative = "overexpression")
```

**Arguments**

states_vs_t	a matrix with time in lines and individual states in columns
alpha	the family-wise error rate in the case of Bonferroni multiple-hypothesis correction, or the false discovery rate in the case of the FDR multiple hypothesis correction
MHT.correction	the type of multiple hypothesis correction
exclude.states	a vector of states to remove from states_vs_t
states.pair.types	accepted value: "all", "same", "different": selects which kind of state pairs are tested: if there are two states e.g. (1,2), "all" allows (1,1), (2,2) and (1,2); "same" allows (1,1) and (2,2), while "different" allows (1,2)
alternative	a string either equal to "overexpression" or "underexpression"

**Value**

a data.table object of the node pairs significantly correlated at the alpha level. The columns are: link\_id: a unique identifier for links; i and j are the node names; si and sj are their states; pv is the value associated to the link; threshold is the Multiple Hypothesis Testing-adjusted p-value threshold (only links with  $pv \leq \text{threshold}$  are kept)

**Examples**

```
x=sample(c(1,0),1000,replace = TRUE)
xx=x; x[1]=1-x[1] # one modifies just one element
x_rev=rev(x)
y=sample(c(1,0),1000,replace = TRUE)
M=cbind(x,xx,x_rev,y)
mylinks=SVN_links(M)
print(mylinks)
# one can compute clusters as well
myclusters=SVN_clusters(mylinks)
```

# Index

\*Topic **network, synchronization**

SVN-package, [2](#)

SVN (SVN-package), [2](#)

SVN-package, [2](#)

SVN\_clusters, [3](#)

SVN\_links, [3](#)